Final Project Report for Song Database

Designing a database and application for a collection of songs

Alan Zhan and Samantha Tuberman
CS3200, section 3

Project Video
https://youtu.be/nSXx_R6OcJE

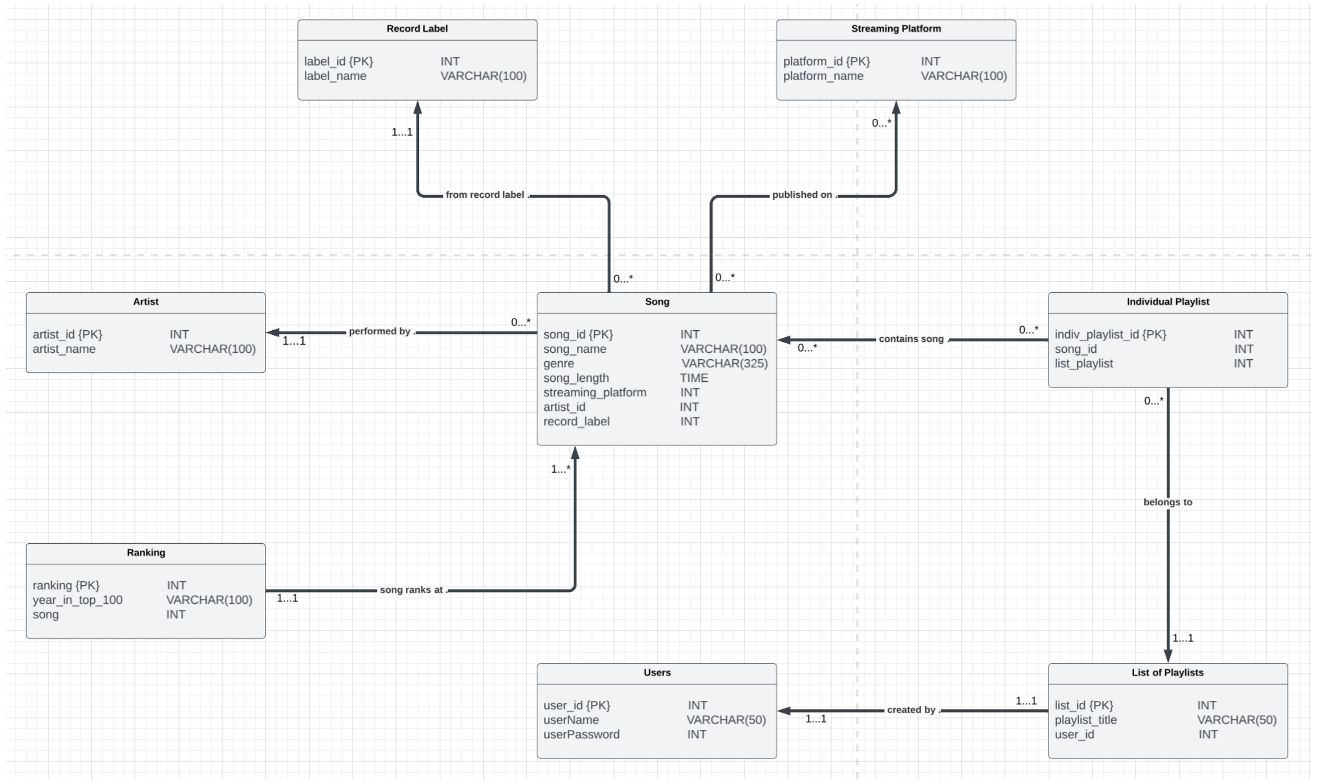
README - specifications for building the project on another computer

1. Import the songdb database dump into your mysql database
2. Go into the my-app folder
3. Run the command 'npm install' to install all dependencies
4. Open the index.js file and change the values under the comment to match your username, password, and database name
5. Run the command "npm run start" to start the website on localhost:3000
6. Run the command "node index.js" to start the backend on on localhost:3001
7. Go to localhost:3000 to see the website (page may take a while to startup)

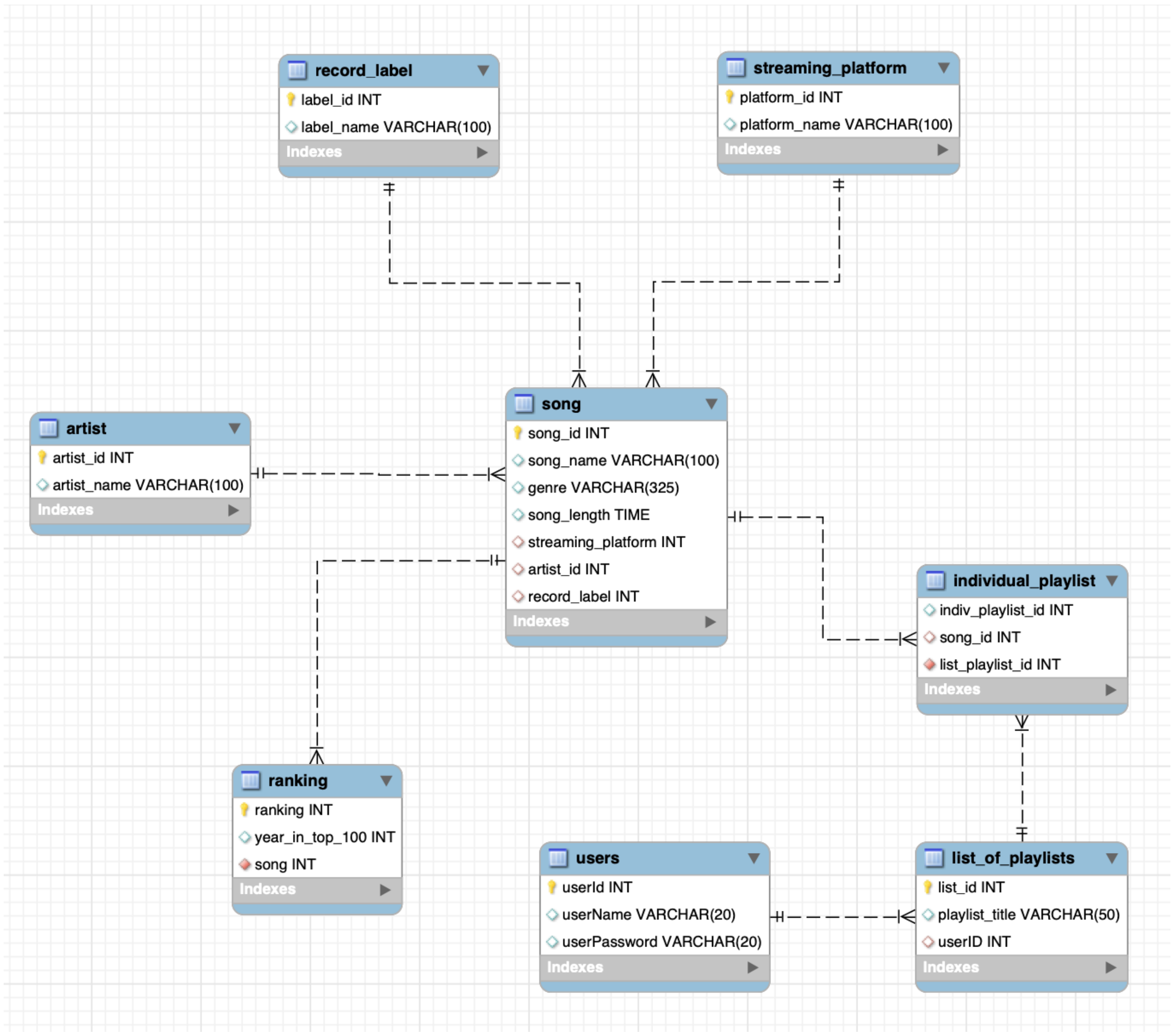
Technical Specifications

This project was built using the host language **javascript** using **Node.js** and **React**, and was connected to **SQL**, which contained the database itself. UI elements from the material-ui library were used as well.

Conceptual Design of Database - UML Diagram



Logical Design of Database



Final User Flow of the Database System

- User opens application
- User logs into application
    - If user already has an account, the user types in their username and password to log into the application
    - If the user does not have an account, they click the “create account” button to make a new account
        - User then logs into system
- User can search for songs
    - User can filter the search by song title, artist, genre, etc.
- User can create playlists
    - User adds songs to the playlist and names it
    - User can also edit playlists by adding or deleting songs from said playlist or by changing the playlist name
- User can delete their account
- User logs out of database
-

Lessons Learned

Throughout this project, we gained various technical expertise and insights about building a database application. For instance, we learned to use new coding environments such as Node.js and react in order to build a website as the front-end of our database application. In this, we learned the ways in which Node.js connects to and interacts with a SQL database. This then allowed us to learn how to use an external server like Node.js to manipulate the database in order to contribute to the final application. Ultimately, we learned how to do backend and frontend design using Node.js to allow for a functional application that had access to a SQL database.

In addition to said technical skills, we also gained insights about how to build a functional database application in a relatively efficient manner. We learned that using a UI library when coding the application can save time as it allows one to make use of premade elements. This allows one to avoid designing all elements of the application from scratch as that can be very time consuming. Additionally, we learned that it is essential to divide work between group members based on what each member is best at, while still working to support each other and answer questions the other members may have. This strategy can also help to avoid unnecessary loss of time.

When starting this project, we anticipated that we were going to use Java and Swift as a means of creating the GUI for the project. However, once we began working on it, we decided to use Node.js and react instead as they allowed us to make a GUI with more ease. This is an example of an alternative strategy that we realized and adjusted to while working on this project.

Future Work for this Database

This database is intended to store a collection of the most popular songs according to a particular ranking system. It is intended to allow users to discover these songs and view various details about them, including their genre, the artist that performed the song, the record label that produced the song, and more. In addition to users being able to search for songs based on different criteria, this database is intended to allow users to create playlists of songs they enjoy.

In the future, there are several additional functionalities that can be added to this database to make the user experience more comprehensive. For instance, the ability of users to share playlists they have made with other users would allow for a more social aspect to this database. It would also be interesting to add a functionality for users to coauthor playlists, creating a blend of two users' music choices. Other simple additions to this database include adding more information about individual songs and artists, including the album a song belongs to, other albums the artist has written and performed, and songs similar to those added to playlists based on their genre. Finally, a second user type, manager, can be added to the database to allow for permanent changes to the database. As of now, users are able to add and delete songs from playlists they create. A manager would have the additional ability to add and delete songs from the entire database, making it more dynamic.